

DIALOG(R)File 347:JAPIO
(c) 2005 JPO & JAPIO. All rts. reserv.

02788672 **Image available**
INDIRECT ADDRESS SYSTEM

PUB. NO.: 01-086272 [JP 1086272 A]
PUBLISHED: March 30, 1989 (19890330)
INVENTOR(s): MANABE TOSHIHIKO
APPLICANT(s): TOSHIBA CORP [000307] (A Japanese Company or Corporation), JP
(Japan)
APPL. NO.: 62-167037 [JP 87167037]
FILED: July 06, 1987 (19870706)
INTL CLASS: [4] G06F-015/347
JAPIO CLASS: 45.4 (INFORMATION PROCESSING -- Computer Applications)
JOURNAL: Section: P, Section No. 900, Vol. 13, No. 316, Pg. 96, July
18, 1989 (19890718)

ABSTRACT

PURPOSE: To widely economize a memory by equipping the storing means of an index value and the complementary generating means of the index value and computing the element address of a designated vector from the index value.

CONSTITUTION: The title system is composed of a storage 1 to store the aggregation of the index value, a complementary generating part 2 to generate a complementary to the aggregation of the index value which is read from the storage 1 and an address generating part 3 to generate the address of data from the generated index value. The storage 1 is addressed to the word unit of a constant length and it stores the aggregation of the index in the ascending order of power. The top word of a storing area stores the maximum value of the index to be a subject. The second word stores the top index value. After the third word, difference in the preceding index value is stored.

?

⑫ 公開特許公報(A)

昭64-86272

⑤ Int.Cl.⁴

識別記号

庁内整理番号

④ 公開 昭和64年(1989)3月30日

G 06 F 15/347

B-7056-5B

審査請求 未請求 発明の数 1 (全7頁)

⑭ 発明の名称 間接アドレス方式

⑯ 特 願 昭62-167037

⑰ 出 願 昭62(1987)7月6日

⑱ 発 明 者 真 鍋 俊 彦 神奈川県川崎市幸区小向東芝町1 株式会社東芝総合研究所内

⑲ 出 願 人 株 式 会 社 東 芝 神奈川県川崎市幸区堀川町72番地

⑳ 代 理 人 弁 理 士 則 近 憲 佑 外1名

明 細 書

1. 発明の名称

間接アドレス方式

2. 特許請求の範囲

ベクトルまたはマトリックスの中で演算対象となっている要素の相対的な位置を示すインデックス値の集合を複数組格納できる記憶装置と、この記憶装置からインデックス値の集合を読み出してきて指定ベクトルまたはマトリックスに対して読み出された集合に含まれないインデックス値を生成する生成手段と、この生成手段により生成されたインデックス値から指定ベクトルまたはマトリックスの要素のメモリ中のアドレスを計算する手段を有することを特徴とする間接アドレス方式。

3. 発明の詳細な説明

〔発明の目的〕

(産業上の利用分野)

本発明は科学技術計算でしばしば必要とされるベクトルやマトリックスの計算を高速に行なう間接アドレス方式に関する。

(従来技術)

科学技術計算では少く規模が大きくなるとベクトルやマトリックスに対する計算を必要とし、計算時間がかかる。これらのベクトルやマトリックスの各要素は、通常、計算機のメモリの中で、そのインデックスの順に一定のアドレス間隔で格納されることが多い。そこでこのようなデータの集合を“配列”と呼ぶ。ベクトルプロセッサ(またはアレイプロセッサ)はこのような、一定のアドレス間隔で格納されているデータに対して、パイプライン技術を用いて、連続的に高速に同一種類の演算を行なうように構成されており、これによって、配列に対する計算のスピードをあげようとするものである。すなわち従来のベクトルプロセッサは次のような計算を高速に実行することができる。

<例1>

D0 10 I=1,100

A(I)=B(I)*C(I)

10 CONTINUE

しかし、実際の応用では次のような要素毎に計算内容が異なる場合がかなり頻繁に現われる。

<例2>

```
DO 20 I=1,100
  IF(B(I).GT.0.0) THEN    ...条件1
    A(I)=A(I)+C(I)        ...文1
  ELSE
    A(I)=A(I)*D(I)        ...文2
  ENDIF
20  CONTINUE
```

このような場合に対処するため、間接アドレスによるベクトル演算を行なえるベクトルプロセッサが開発されてきている。このようなベクトルプロセッサでは、上記の例(例2)に対して、最初に、次の二種類のインデックスの集合IX, IYを生成する。次に、文1に対してはIX内のインデックスを、文2に対してはIY内のインデックスを元に演算を行なっていく。

```
IX = { I | B(I) > 0.0 }
IY = { I | B(I) <= 0.0 }
```

- 3 -

領域1と領域2は、それぞれ配列Bの要素数分の大きさである。領域3はIX内のインデックスが、領域4はIY内のインデックスが実際に格納された領域であり、領域5と領域6は余った領域である。領域3と領域6、領域4と領域5は同じ大きさである。領域2の後に続く領域7があていれば、領域6は領域7と合せて他のインデックス生成に利用できるが、領域5は他に利用することが難しく、無駄な領域として残ってしまう。

(発明が解決しようとする問題点)

このように、ベクトルやマトリックスの要素を対象とした演算では、インデックスを基本とした間接アドレス方式によるベクトル演算が有効であるが、従来の間接アドレス方式では、メモリを大量に消費するという欠点をもっていた。特に、インデックス生成の際には、無駄な領域が生じ、メモリの有効活用の点から見て望ましくなかった。

この発明は、上記の問題点に鑑み、インデックスを生成するための領域が必要最少限で済み、インデックス生成の際にも無駄な領域を生じない間

配列のインデックスを基本とする間接アドレス機構をもてば、このように、計算内容が異なる場合に対しても比較的効率よくベクトル演算を行なえる。しかし、インデックスの集合をすべて記憶しておかなくてはならないので、メモリを大幅に消費するという欠点も併せもっている。

例えば、上記のIXとIYの和集合の要素数は、配列Bの要素数と等しい。したがって、Bの大きさが大きい程、インデックスの集合のためにメモリを消費することになる。また、インデックス生成の際に、配列の要素数の二倍のインデックスを格納できる領域を用意しておかなくてはならない。これは、インデックス生成時には、上記のIXとIYに含まれるインデックスの個数が分らないため、それらの両方に対して配列の要素数分のメモリ領域を確保しておかなくてはならないためである。さらに、IXとIYの領域をメモリ上で連続して割当てると、IXとIYの間に無駄な領域が生じる。この様子を第6図を使って説明する。領域1はIX用に、領域2はIY用に割当てられたメモリ領域である。

- 4 -

接アドレス方式を提供することを目的とする。

(発明の構成)

(問題点を解決するための手段)

この発明は、インデックスを基本とする間接アドレス方式において、インデックス値の集合を記憶する記憶装置と、既記憶装置からインデックス値の集合を読み出してきて、指定ベクトルまたはマトリックスに対してのインデックス値の補集合を生成する手段と、既手段によって生成されたインデックス値から指定ベクトルまたはマトリックスの要素のアドレスを計算する手段を有することを特徴とする。

(作用)

この発明を利用すれば、前記の例2のような場合に、IXだけを記憶して文1と文2の両方について、間接アドレスによるベクトル演算を行なうことができる。最初に、文1については、IXを利用して、通常の間接アドレス方式のベクトル演算で実行する。次に、文2についてはこの発明を利用して、記憶装置からIX内のインデックス値を読み

出してきて、IXの補集合IYを生成し、IY内のインデックス値に従ってベクトル演算を実行する。同様に、IYだけを記憶して文1と文2の両方を実行でき、その際には文1について、この発明が適用される。すなわち、例2のような場合に、片方のインデックス値の集合だけを記憶して、文1と文2の両方を間接アドレス方式で実行できるようになる。

(実施例)

以下、図面を用いてこの発明の実施例を説明する。

第1図はこの発明の一実施例に係わる間接アドレス方式のブロック図である。同図に示す間接アドレス方式は、インデックス値の集合を記憶する記憶装置1と、記憶装置1から読み出してきたインデックス値の集合に対して補集合を生成する補集合生成部2と、生成されたインデックス値からデータのアドレスを生成するアドレス生成部3から成る。本実施例では、1からNまでのインデックスについての部分集合から、その補集合にあた

るインデックス値による、データのアドレスを生成する。

記憶装置1は、一定長の語単位にアドレス付けられていて、次に説明する形で昇べきの順でインデックスの集合を記憶している。集合が記憶されている記憶領域の先頭の語は、対象となる配列のインデックスの最大値を格納している。2番目の語は、先頭のインデックス値を格納している。3番目以降の語は、その前のインデックス値からの差を格納している。そして、記憶領域の最後の語は、区切りとして負の値を格納している。

第2図はインデックス値の集合が格納されている領域の内容の例である。1000番地から1006番地までインデックス値の一つの集合に関する情報が格納されている。先頭の語である1000番地には10が記憶され、1から10までの連続した値の集合の部分集合が格納されていることが示されている。2番目の1001番地には、部分集合の先頭のインデックス値である3が格納されている。1002番地から1005番地までは、この部分集合に属する各イン

- 7 -

デックス値とその一つ前のインデックス値との差が格納されている。最後の1006番地には区切りとして-1が格納されている。したがって、第2図の例の部分集合は5個のインデックス値から成り、その内容は次の通りである。

(3, 5, 6, 7, 9)

第3図は本実施例の補集合生成部の詳細なブロック図である。レジスタLIMIT4にはインデックスの最大値が設定される。レジスタINDEX5は生成中の補集合のインデックス値を保持するためのレジスタで、初期値は1に設定され、インクリメンタ12によりカウントアップの動作が行なわれる。レジスタINDEXの値は信号IDXに出力される。レジスタLIMITとレジスタINDEXの値が同じになれば、コンパレータ15により信号GONがHとなる。これは、生成したインデックス値とインデックスの最大値が等しくなったので、補集合の生成が終了したことを示している。レジスタDISP6は、記憶装置1の中に記憶されているインデックス値の集合での、隣り

- 8 -

あったインデックス値の差が格納される。レジスタCOUNT7は初期値が1に設定され、レジスタDISPの値と同じになるまでレジスタINDEXと同期して、インクリメンタ13によりカウントアップの動作が行なわれる。レジスタDISPとレジスタCOUNTの値が等しくなれば、コンパレータ16により信号RDYNがHとなり、カウントアップが終了し、再びレジスタCOUNTに1が設定される。信号RDYNがLの間、信号IDXには補集合のインデックス値が出力されている。レジスタADDR8は、記憶装置1内に第2図に示す形式で格納されたインデックス値の集合を読み出すアドレスを格納するものであり、最初にインデックス値の集合が記憶されている領域の先頭アドレスSAが初期設定される。そして、信号RDYNがHとなったときにインクリメンタ14によりカウントアップされ、そのときの信号IADDRの値をアドレスとして記憶1からデータを信号DATAに読み出してくる。この読み出された値がレジスタDISPに格納される。なお、カウ

トアップと信号 I A D R による記憶装置 1 へアクセスは、対象としている記憶領域の 2 語目を読み出してくるまでの初期化シーケンスの中では、信号 R D Y N に係わらず行なわれる。

第 4 図は、第 2 図の形で格納されたインデックス値の集合に対して補集合生成部 2 を動作させたときに、第 3 図の各レジスタと信号 R D Y M の値を、同期信号の周期を単位とした時刻に従って示したものである。第 4 図を使って補集合生成部 2 の動作例を説明する。時刻 0 から 3 までは初期化シーケンスである。時刻 1 ではレジスタ I N D E X に初期値 1 が、レジスタ A D R に集合の記憶領域の先頭アドレスである 1000 が設定される。時刻 2 ではレジスタ A D R の値に従って、1000 番地の内容が読み出され、レジスタ L I M I T に設定される。同時にレジスタ A D R の内容がカウントアップされる。時刻 3 ではレジスタ A D R の内容に従って、1001 番地の内容が読み出され、今度はレジスタ D I S P に設定される。同時に、レジスタ C O U N T に初期値 1 が設定され、レジスタ A D

R の内容がカウントアップされる。時刻 3 で初期化シーケンスを終了し、第 3 図の全レジスタに値が設定された。時刻 4 以降では信号 G O N が H になるまで、レジスタ I N D E X は毎時刻カウントアップする。レジスタ C O U N T は信号 R D Y N が L のときにカウントアップし、H のときに 1 が設定される。また、レジスタ A D R は信号 R D Y N が H のときだけカウントアップされる。信号 R D Y N が H になっているのは、レジスタ D I S P とレジスタ C O U N T の値が等しくなった時刻 5, 7, 8, 9, 11 である。その他では、時刻 3 以降信号 R D Y N は L になっている。信号 R D Y N が L になっているときには、記憶装置 1 に格納されたインデックス値の集合のとなりあった要素の中間の値がレジスタ I N D E X 上にあるので、信号 I D X には補集合のインデックス値が出力されている。信号 R D Y N が L になっているのは、時刻 3, 4, 6, 10, 12 で、そのときのレジスタ I N D E X の値をまとめると、

(1, 2, 4, 8, 10)

- 11 -

となり、第 2 図の形式で表現されている集合の補集合となっていることが確認できる。信号 G O N は、レジスタ L I M I T とレジスタ I N D E X の値が等しくなった時刻 12 で H になり、このとき、補集合生成部の動作は終了する。

第 5 図は、アドレス生成部のブロック図である。レジスタ I 17 はアドレス生成の基本となるインデックス値を記憶するためのレジスタで、第 3 図の信号 R D Y N が L のときに、デクリメンタ 21 を通して信号 I D X の値から 1 引いた値が設定される。ここで 1 減ずるのはインデックスが 1 オリジン (1 から始まっている) であるとしているためである。レジスタ U N I T 18 にはインデックス値が 1 増加するアドレスの増分が、補集合生成部 2 の初期化シーケンスの間に設定される。各要素の大きさを 4 (4 バイト) とすると、一次元配列 (A (100) のような配列) では U N I T = 4、二次元配列 (B (100, 100) のような配列) で列方向にアクセスする場合には、U N I T = 400 が格納される。レジスタ B A S E 19 にはインデックスの対象とな

っている配列の先頭アドレスが、同様に初期化シーケンスの間に設定される。レジスタ U N I T と B A S E の値はアドレス生成中には変わらない。レジスタ E N A B L E 20 は 1 ビットのレジスタで、1 サイクル前の信号 R D Y N の値を記憶している。レジスタ E N A B L E の値は信号 E N B に出力されている。信号 D A D R には、レジスタ I にレジスタ U N I T の値を乗じた結果に、レジスタ B A S E の値を加えた結果が、乗算器 22 と加算器 23 を通して出力される。すなわち、信号 D A D R には、

$$DADR = (IDX - 1) \times UNIT + BASE$$

が出力される。信号 E N B が L のときに、信号 D A D R 上に補集合のインデックスを元にしたアドレスが生成されている。例えば、各要素の大きさが 4 で、先頭アドレスが 2000 の配列に対して、上記の補集合からは、2000, 2004, 2012, 2028, 2036 の五つのアドレスが生成される。

なお第 2 図で示したインデックス値そのもの (補集合でないもの) を発生する回路は、第 2 図の 1001 で示した値に、1002 以降の値を順に加算す

るだけで求められる。その回路構成は明白であるのでここでは述べない。

このように、本実施例によれば、インデックスの部分集合から、その補集合のインデックスに対応するアドレスを生成でき、インデックスを格納するための記憶容量を節約できる。さらに本実施例では、記憶装置1の中で、インデックスの値そのままでなく、その差を格納しているので、一個のインデックスに必要な容量も少なく済む。その上、記憶装置1にインデックス値をそのままの形で格納する場合にも少しの修正で対応できる。
〔発明の効果〕

以上、説明したように、本実施例によれば、

(1) 記憶装置内に格納しておかなくてはならないインデックスの集合を、例2のような場合に、IXとIVのどちらかに限定できるので、実際にインデックスが格納される領域の容量を減少させることができる。

(2) 例2でのIXまたはIVといったインデックスの部分集合を生成する際に、予め確保しておかな

くてはならない記憶容量が、基本となる配列の要素分で済み、本発明を適用しない場合の半分で済む。

(3) 生成したインデックスの部分集合の間に無駄な領域を生じることはない。これは、本発明を利用すれば、例2のような場合に、IXとIVのどちらか一方だけを生成すればよいので、第6図のIXとIVの領域の間にはさまれた領域5が生じないためである。

(4) 複雑な構成を必要としないで、(1)〜(3)を実現でき、その効果の組合せにより、本発明を適用しない場合に比べて、大幅にメモリを節約できる。

4. 図面の簡単な説明

第1図は本発明の一実施例の構成を示すブロック図、第2図は第1図の記憶装置に格納されたインデックスの集合の格納形式を示す図、第3図は第1図の補集合生成部の構成を示したブロック図、第4図は第3図の構成の補集合生成部の動作例を示した図、第5図は第1図のアドレス生成部の構成を示したブロック図、第6図は従来の間接アド

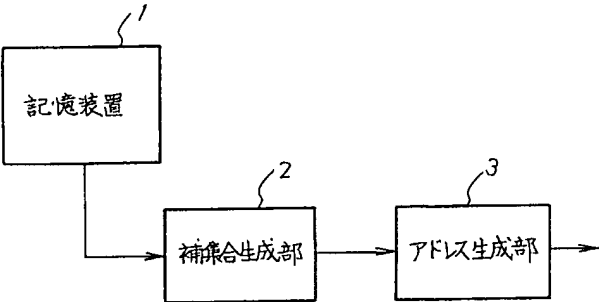
- 15 -

- 16 -

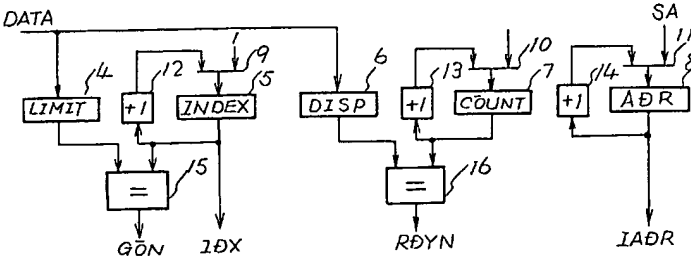
レス方式によるインデックスの部分集合の格納形式を示す図である。

- 1…記憶装置、 2…補集合生成部、
- 3…アドレス生成部、
- 4, 5, 6, 7, 8, 17, 18, 19, 20…レジスタ、
- 9, 10, 11…マルチプレクサ、
- 12, 13, 14…インクリメンタ、
- 15, 16…コンパレータ、 21…デクリメンタ、
- 22…乗算器、 23…加算器。

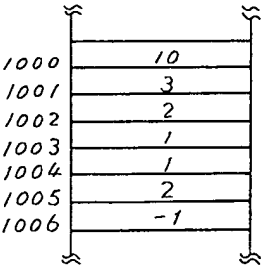
代理人 弁理士 則 近 憲 佑
同 松 山 允 之



第 1 図



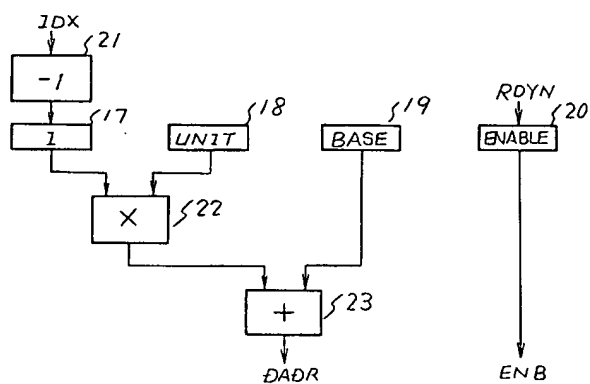
第 3 図



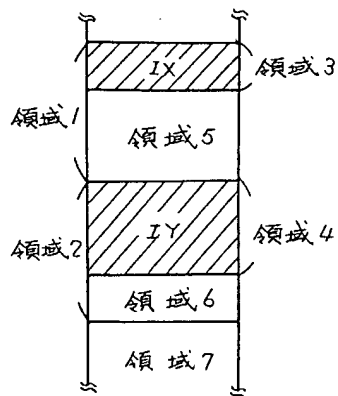
第 2 図

時刻	LIMIT	INDEX	DISP	COUNT	ADDR	RBYN
0	—	—	—	—	—	—
1	—	1	—	—	1000	—
2	10	1	—	—	1001	—
3	10	1	3	1	1002	L
4	10	2	3	2	1002	L
5	10	3	3	3	1002	H
6	10	4	2	1	1003	L
7	10	5	2	2	1003	H
8	10	6	1	1	1004	H
9	10	7	1	1	1005	H
10	10	8	2	1	1006	L
11	10	9	2	2	1006	H
12	10	10	-1	1	1007	L

第 4 図



第 5 図



第 6 図